BUTTERWORTH HEINEMANN

Grid generation in two dimensions using the complex variable boundary element method

Robert T. Bailey

Science Applications International Corporation, Abingdon, MD, USA

C. K. Hsieh and H. Li

Department of Mechanical Engineering, University of Florida, Gainesville, FL, USA

An iterative variation of the linear-element complex variable boundary element method (CVBEM) is used to generate grid points in two-dimensional simply connected spatial domains. As in many grid generation techniques, the solution of Laplace's equation is involved; however, the boundary element formulation results in a much smaller set of simultaneous equations, and once the values of the complex potential at the boundaries are fully determined, the internal grid point distribution can be altered without resolving the elliptic system. The boundary conditions implemented produce grid line orthogonality at the domain boundaries. Additionally, by using complex variables in the formulation, numerical integration is avoided. The utility of the method is demonstrated by (1) generating grid systems for several irregular geometries and (2) obtaining a numerical solution for the flow field associated with one of the geometries.

Keywords: CVBEM, numerical grid generation, boundary elements

1. Introduction

Finite-difference methods (FDMs) have proven to be an effective tool for obtaining solutions to partial differential equations (PDEs), especially in the field of computational fluid dynamics (CFD).[1,2] All FDMs require that the continuous domain of interest be replaced by a discrete domain composed of a collection of points distributed throughout the original domain (discretized). These points are known as grid points, and the collection of grid points is known as a grid system. Proper placement of these grid points is essential if accurate solutions are to be obtained. Often, nonuniform grid-point distributions are necessary in order to accommodate irregular geometries or to accurately resolve complex flow conditions. Unsteady problems may even require grid points that move over time. Unfortunately, it is extremely difficult and impractical to derive finite-difference equations (FDEs) using nonuniformly distributed and moving grid points. For this reason, the grid points in the "physical" spatial domain are mapped onto a "transformed" computational domain where they are stationary and uniformly distributed (see *Figure 1*). This process is known as grid generation.

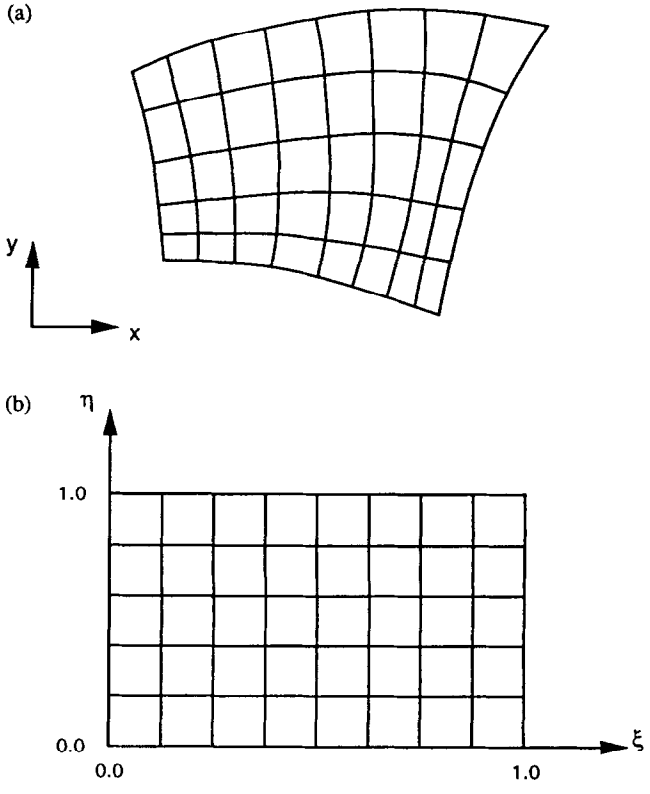

Figure 1. The (a) physical and (b) computational domains used in numerical grid generation.

Address reprint requests to Dr. Robert T. Bailey at Science Applications International Corp., 1309 Continental Drive, Suite F, Abingdon, MD 21009, USA.

Received 25 September 1992; revised 5 December 1994; accepted 16 January 1995

Appl. Math. Modelling 1995, Vol. 19, June

655 Avenue of the Americas, New York, NY 10010

0307-904X/95/$10.00
SSDI 0307-904X(95)00008-8

Methods for performing grid generation are traditionally divided into two major classes: differential equation methods (including conformal mappings) and algebraic methods. Differential equation methods determine grid point locations by solving one or more PDEs that describe the transformation between the physical spatial domain and the computational domain. This usually requires a significant computational effort, but these methods can produce grid systems whose grid lines are smooth and nonoverlapping. On the other hand, algebraic methods generate grid systems by interpolating between the boundaries of the physical spatial domain. Since algebraic methods do not involve the solution of PDEs during the grid generation process, they tend to be much more computationally efficient than differential equation methods. However, grid systems generated by algebraic methods may have grid lines that are nonsmooth and overlapping, characteristics that must be corrected if meaningful finite-difference solutions are to be obtained using such grids.

Interestingly enough, most differential equation methods generate solutions for their mappings by using FDMs (a peculiarly circular arrangement). The PDEs may be parabolic, hyperbolic, or, most often, elliptic. Indeed, elliptic grid generation methods (first proposed by Winslow[3]) have become quite refined due to the work of Thompson et al.[4–7] and numerous other investigators. Other methods (besides FDMs) for solving elliptic PDEs certainly exist, and some of the more powerful are the boundary element methods (BEMs), also known as integral equation methods.[8,9] Unlike FDMs, which require that the entire domain be discretized, BEMs only involve discretization of the boundaries of the domain. In this way, BEMs effectively reduce the dimension of the discretization by one, and, in the process, significantly decrease the number of simultaneous equations that need to be solved. These benefits do not come free, though, as the matrices associated with these systems of equations tend to be nearly fully populated in BEMs, while they are comparably sparse in implicit FDMs.

Thompson et al.[4–6] discuss the use of integral equation methods in the numerical construction of conformal mappings, with the implication that such methods could be used for numerical grid generation. However, it is mentioned that traditional application of such methods actually produces the inverse of the mapping desired for grid generation and that additional auxiliary transformations may also be necessary. It appears that the only application of a BEM specifically for numerical grid generation is due to Tsay.[10] In that paper, the author describes a method for generating grid systems for four-sided, two-dimensional (2D), simply connected spatial domains where two of the sides are limited to straight lines. Due to the Dirichlet boundary conditions imposed, the grids generated were not orthogonal. (Generally, it is desirable that grid lines of opposite families intersect both each other and the domain boundaries orthogonally.[4])

The BEM used by Tsay was formulated using real variables (although a complex function was introduced to improve efficiency in the implementation). If one instead begins the boundary element formulation with complex variables, the so-called complex variable boundary element method (CVBEM) results.[11] The CVBEM is limited in application to Laplace's and Poisson's equations in two dimensions, but it does possess the following two significant advantages over real variable boundry element methods (RVBEMs): (1) the expressions for calculating values of the potential at points on the interior of the domain are analytic and satisfy exactly the 2D Laplace equation and (2) all integrations are carried out analytically without the need for numerical integration. These two factors combine to give the CVBEM high accuracy and efficiency.

In this paper, an application of the CVBEM to numerical grid generation in 2D simply connected spatial domains is presented. The CVBEM has never been applied in this area, but since it involves the solution of the Laplace equation, it is properly classified as a differential equation method (although the term integral equation method is more appropriate). The utility and flexibility of the method make it a powerful addition to the selection of grid generation methods currently available. Before the application to grid generation is described, however, the fundamentals of the linear-element CVBEM are presented.

2. Description of the linear-element CVBEM

The formulation of the linear-element CVBEM begins with Cauchy's integral formula,

$$\omega(z_0) = \frac{1}{2\pi i}\int_{\Gamma} \frac{\omega(\zeta)}{\zeta - z_0}\, d\zeta, \qquad z_0 \in \Omega, \qquad z_0 \notin \Gamma \tag{1}$$

This expression relates the value of the complex potential, $\omega = \phi + i\psi$, at point z_0 located inside the k-connected Jordan domain, Ω, to a contour integral (containing ω) along the boundary, Γ. The direction of travel for the contour integral is such that the interior of the domain is always to the left. In the complex potential, ϕ refers to the real potential, while ψ refers to the stream function.

The linear-element formulation of the CVBEM transforms the Cauchy integral formula into a BEM by using two major approximations. First, boundary Γ is discretized into N finite-length segments (elements), Γ_j, the end points of which are referred to as nodal points or nodes. The domain boundary is then taken as the union of these elements as shown in *Figure 2*, i.e.,

$$\Gamma = \bigcup_{j=1}^{N} \Gamma_j \tag{2}$$

Second, the function $\omega(z)$ is approximated by a linear global trial function, $G_1(z)$, given by

$$G_1(z) = \sum_{j=1}^{N} N_j(z)\omega_j \tag{3}$$

where ω_j is the value of ω at nodal point z_j, and $N_j(z)$ is a continuous basis function weighting the effect of ω_j over elements Γ_{j-1} and Γ_j. This basis function is taken

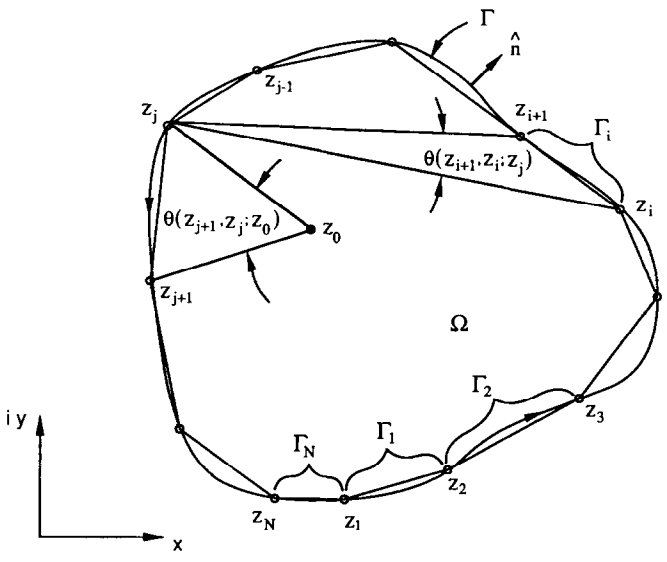

Figure 2. Boundary discretization and angle definitions in the linear-element CVBEM.

as a first-degree polynomial of the following form:

$$N_j(z) = \begin{cases} (z - z_{j-1})/(z_j - z_{j-1}) & z \in \Gamma_{j-1} \\ (z_{j+1} - z)/(z_{j+1} - z_j) & z \in \Gamma_j \\ 0 & z \notin \Gamma_{j-1} \cup \Gamma_j \end{cases}$$

It is necessary to explain the notation that will be used throughout the remainder of this paper. The subscript "exact" will identify a quantity whose value is known exactly. The overbar will refer to a quantity whose value is specified, such as in the boundary conditions. Finally, the hat will represent a quantity whose value is treated as an unknown in the solution by the CVBEM. Quantities with such designations missing should be assumed to represent a general case in which the quantity in question may be either specified or unknown depending on the circumstances.

By substituting $G_1(\zeta)$ for $\omega(\zeta)$ in the right-hand side of the Cauchy integral formula (equation [1]), the first-order CVBEM approximation of ω can be expressed as

$$\hat{\omega}(z_0) = \frac{1}{2\pi i} \int_\Gamma \frac{G_1(\zeta)}{\zeta - z_0}\, d\zeta, \qquad z_0 \in \Omega, \qquad z_0 \notin \Gamma \tag{4}$$

After much operation (see Ref. 11), the contour integration can be carried out, and equation (4) reduces to

$$\hat{\omega}(z_0) = \frac{1}{2\pi i} \sum_{j=1}^{N} [\omega_{j+1}(z_0 - z_j) - \omega_j(z_0 - z_{j+1})] \frac{h_j}{(z_{j+1} - z_j)} \tag{5a}$$

where

$$h_j = \ln\left[\frac{(z_{j+1} - z_0)}{(z_j - z_0)}\right] = \ln\left|\frac{(z_{j+1} - z_0)}{(z_j - z_0)}\right| + i\theta(z_{j+1}, z_j; z_0) \tag{5b}$$

This formula forms the basis for the linear-element CVBEM. (See *Figure 2* for the definition of $\theta[z_{j+1}, z_j; z_0]$).

Equation (5), being expressed in complex variables, actually embodies two equations: one for the real part and one for the imaginary part. If the values of ϕ and ψ (and thus ω) are known at each boundary node, equation (5) can be used to calculate $\hat{\phi}$ and $\hat{\psi}$ at any interior point, z_0. In most potential problems, however, boundary conditions specify either ϕ, ψ, or neither of them explicitly. To solve for the unknown values of ϕ and ψ, it is necessary to derive an extended version of equation (5) by moving z_0 to the position of z_j on the boundary. In this effort, one cannot simply plug in z_j for z_0 in equation (5) because $(z_j - z_0)$ appears in the denominator of the natural log term. Instead, one takes the limit of equation (4) as z_0 approaches z_j. Hromadka[11] has performed this somewhat involved task, with the result

$$\hat{\omega}_j = \frac{1}{2\pi i}\left\{\omega_j\left[\ln\left|\frac{z_{j+1} - z_j}{z_{j-1} - z_j}\right| + i\theta(z_{j+1}, z_{j-1}; z_j)\right] + \sum_{\substack{i=1 \\ i,i+1 \neq j}}^{N} [\omega_{i+1}(z_j - z_i) - \omega_i(z_j - z_{i+1})]h_i/(z_{i+1} - z_i)\right\} \tag{6a}$$

where

$$h_i = \ln\left[\frac{z_{i+1} - z_j}{z_i - z_j}\right] = \ln\left|\frac{(z_{i+1} - z_j)}{(z_i - z_j)}\right| + i\theta(z_{i+1}, z_i; z_j) \tag{6b}$$

The angles $\theta(z_{j+1}, z_{j-1}; z_j)$ and $\theta(z_{i+1}, z_i; z_j)$ are shown in *Figure 2*.

Equation (6) can be applied at any boundary node, but like equation (5), equation (6) has real and imaginary parts. Two equations can thus be derived for arbitrary boundary node j as

$$\hat{\phi}_j = \frac{1}{2\pi}\left\{\psi_j \ln\left|\frac{z_{j+1} - z_j}{z_{j-1} - z_j}\right| + \phi_j\theta(z_{j+1}, z_{j-1}; z_j) + \sum_{\substack{i=1 \\ i,i+1 \neq j}}^{N} \{\phi_{i+1}C_2 + \psi_{i+1}C_1 - \phi_i C_4 - \psi_i C_3\}\right\} \tag{7a}$$

and

$$\hat{\psi}_j = -\frac{1}{2\pi}\left\{\phi_j \ln\left|\frac{z_{j+1} - z_j}{z_{j-1} - z_j}\right| - \psi_j\theta(z_{j+1}, z_{j-1}; z_j) + \sum_{\substack{i=1 \\ i,i+1 \neq j}}^{N} \{\phi_{i+1}C_1 - \psi_{i+1}C_2 - \phi_i C_3 + \psi_i C_4\}\right\} \tag{7b}$$

where

$$C_1 = [(x_j - x_i)C - (y_j - y_i)D] \tag{7c}$$

$$C_2 = [(x_j - x_i)D + (y_j - y_i)C] \tag{7d}$$

$$C_3 = [(x_j - x_{i+1})C - (y_j - y_{i+1})D] \quad (7e)$$

$$C_4 = [(x_j - x_{i+1})D + (y_j - y_{i+1})C] \quad (7f)$$

$$C = [A(x_{i+1} - x_i) + B(y_{i+1} - y_i)]/F \quad (7g)$$

$$D = [B(x_{i+1} - x_i) - A(y_{i+1} - y_i)]/F \quad (7h)$$

$$F = (x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 \quad (7i)$$

$$A = \ln\left|\frac{z_{i+1} - z_j}{z_i - z_j}\right| \quad (7j)$$

$$B = \theta(z_{i+1}, z_i; z_j) \quad (7k)$$

Henceforth, equations (7a) and (7b) will be referred to as the phi nodal and psi nodal equations, respectively.

2.1 *Boundary conditions*

Having generated the desired equations for calculating $\hat{\phi}$ and $\hat{\psi}$ at the boundary nodes, a question arises as to how one uses equations (7a) and (7b) to solve for the unknown boundary values of ϕ and ψ. Before this question can be answered, consideration must be given to the conditions that exist at the boundary nodes. Dirichlet and Neumann boundary conditions are implemented as detailed below.[12]

Dirichlet condition: The potential at node j is known and specified, i.e.,

$$\bar{\phi}_j = \phi_{\text{exact},j} \quad (8)$$

Notice that, for such a condition, ψ_j is unknown. In keeping with the notation defined previously, the specified ϕ_j becomes $\bar{\phi}_j$, and the unknown ψ_j becomes $\hat{\psi}_j$.

Neumann condition: The normal gradient of the potential at node j is known. The stream function there is then related to the normal gradient of the potential as

$$\hat{\psi}_j = \psi_{j-1} + \frac{1}{2}\left\{\left(\frac{\partial\phi}{\partial n}\right)_j + \left(\frac{\partial\phi}{\partial n}\right)_{j-1}\right\}|z_j - z_{j-1}| \quad (9)$$

Here, it is assumed that $(\partial\phi/\partial n)$ varies linearly from node $j-1$ to node j, as in Refs. 12 and 13. For this condition, both ϕ_j and ψ_j are unknown and are referred to as $\hat{\phi}_j$ and $\hat{\psi}_j$. The use of complex variables in the CVBEM also allows for the occurrence of the stream function condition as follows.

Stream function condition: The stream function at node j is known and specified, i.e.,

$$\bar{\psi}_j = \psi_{\text{exact},j} \quad (10)$$

For this condition, ϕ_j is unknown. The specified ψ_j is renamed $\bar{\psi}_j$, and the unknown ϕ_j becomes $\hat{\phi}_j$. Other types of boundary conditions can also be handled as described in References 12 through 14.

2.2 *Solution methods*

Equations (7a) and (7b) will be used to estimate the unknown values of ϕ and ψ at the boundary nodes. It is clear from their format that $\hat{\phi}_j$ and $\hat{\psi}_j$ on the left can be calculated by using the known values of ϕ and ψ at all boundary nodes whose indices appear on the right. As discussed previously, some of the values of ϕ and ψ at the boundary nodes are not specified by the boundary conditions. The methods for estimating the unspecified values of ϕ and ψ (designated $\hat{\phi}$ and $\hat{\psi}$) thus hinge on how these quantities are related to the specified quantities ($\bar{\phi}$ and $\bar{\psi}$) and on which of the equations (7a or 7b) is used in the construction of the matrix equation for solution of the problem. At nodal point j where a Dirichlet (ϕ-specified) or stream function (ψ-specified) condition is imposed, there are three methods of solving for the estimated nodal values of $\hat{\phi}_j$ or $\hat{\psi}_j$: the explicit, implicit, and hybrid methods. Of these methods, the implicit method is preferred for reasons detailed in Refs. 11 and 12. The implicit method is described below.

Implicit method: For a Dirichlet condition imposed at node j, ϕ_j is known (as $\bar{\phi}_j$) but ψ_j is not. One therefore sets $\phi_j = \bar{\phi}_j$ and $\psi_j = \hat{\psi}_j$ in equation (7b). The first setting is governed by the fact that a Dirichlet condition is specified; no estimation is thus needed for ϕ_j. The second setting is made in order to estimate the unknown value of ψ_j. Equation (7b) thus becomes

$$\hat{\psi}_j = -\frac{1}{2\pi}\left\{\bar{\phi}_j \ln\left|\frac{z_{j+1} - z_j}{z_{j-1} - z_j}\right| - \hat{\psi}_j\theta(z_{j+1}, z_{j-1}; z_j) + \sum_{\substack{i=1 \\ i,i+1 \neq j}}^{N} [\phi_{i+1}C_1 - \psi_{i+1}C_2 - \phi_i C_3 + \psi_i C_4]\right\} \quad (11)$$

This equation is referred to as the implicit psi nodal equation (implicit because the unknown $\hat{\psi}_j$ appears on both sides), and it is sufficient to use this equation to solve for $\hat{\psi}_j$. Equation (7a) is dropped for this node in the solution.

Similarly, for a stream function condition specified at node j, one sets $\psi_j = \bar{\psi}_j$ and $\phi_j = \hat{\phi}_j$. Equation (7a) is now used as

$$\hat{\phi}_j = \frac{1}{2\pi}\left\{\bar{\psi}_j \ln\left|\frac{z_{j+1} - z_j}{z_{j-1} - z_j}\right| + \hat{\phi}_j\theta(z_{j+1}, z_{j-1}; z_j) + \sum_{\substack{i=1 \\ i,i+1 \neq j}}^{N} [\phi_{i+1}C_2 + \psi_{i+1}C_1 - \phi_i C_4 - \psi_i C_3]\right\} \quad (12)$$

This equation is referred to as the implicit phi nodal equation. Equation (7b) is dropped.

For nodes where a Neumann condition is specified, equation (9) is used in conjunction with the implicit phi nodal equation, equation (12), in the solution of unknowns. By applying equations (9) and (12) at all Neumann nodes, equation (11) at all Dirichlet nodes, and equation (12) at all stream function nodes, a system of simultaneous linear algebraic equations is obtained. This system can be represented in a matrix form as

$$\mathbf{C}\begin{Bmatrix}\hat{\phi}\\ \hat{\psi}\end{Bmatrix} = \mathbf{r} \quad (13)$$

Here, C is a fully-populated square matrix of coefficients on the unknown values of $\hat{\phi}$ and $\hat{\psi}$, and $\mathbf{r}$ is a vector of known constants. The system is readily

solved using Gaussian elimination with scaled partial pivoting. Once the unknown values of $\hat{\phi}$ and $\hat{\psi}$ are found at each boundary node, one can use these values, together with the specified values, $\bar{\phi}$ and $\bar{\psi}$, as the boundary nodal values needed by equation (5) for calculating $\hat{\omega}$ at any interior point.

2.3 *Additional comments and the completely specified node*

It should be mentioned that no matter what boundary conditions exist, one must specify a reference value of ψ at some nodal point along the boundary in order to serve as the constant of integration in equation (1). Numerically speaking, the associated matrices become singular if no ψ value is provided. Placing the reference value at node j, one has

$$\bar{\psi}_j = \psi_{\text{exact},j}$$

The implicit phi nodal equation (12) can then be used to determine the unknown $\hat{\phi}_j$ as was discussed earlier in the description of the implicit method.

If the value of ϕ_{exact} is also known at node j (i.e., a Dirichlet condition), one can specify both $\bar{\phi}$ and $\bar{\psi}$ at this node as

$$\bar{\phi}_j = \phi_{\text{exact},j}$$

and

$$\bar{\psi}_j = \psi_{\text{exact},j}$$

The complex potential ω is thus fully specified, and no nodal equation needs to be applied. This node will be called a "completely specified node".

3. Numerical grid generation using the CVBEM

A methodology for using the linear-element CVBEM, just described, to generate grids for 2D simply connected spatial domains is now presented. First, the interior point equations (represented collectively by equation [5]) are rearranged so that if one knows the values of ϕ and ψ at an interior point, one can solve for the point's location, $z_0 = x_0 + iy_0$, provided that the values of ϕ and ψ at the boundary nodes are known. These "inverted" equations will form an integral part of the CVBEM grid generation method. To begin, equation (5) is split into its real and imaginary parts as

$$\begin{aligned}\hat{\phi}(z_0) = \frac{1}{2\pi}\sum_{j=1}^{N} \{&\phi_{j+1}[(x_0 - x_j)D + (y_0 - y_j)C] \\ &+ \psi_{j+1}[(x_0 - x_j)C - (y_0 - y_j)D] \\ &- \phi_j[(x_0 - x_{j+1})D + (y_0 - y_{j+1})C] \\ &- \psi_j[(x_0 - x_{j+1})C - (y_0 - y_{j+1})D]\} \end{aligned} \quad (14a)$$

and

$$\begin{aligned}\hat{\psi}(z_0) = -\frac{1}{2\pi}\sum_{j=1}^{N} \{&\phi_{j+1}[(x_0 - x_j)C - (y_0 - y_j)D] \\ &- \psi_{j+1}[(x_0 - x_j)D + (y_0 - y_j)C] \\ &- \phi_j[(x_0 - x_{j+1})C - (y_0 - y_{j+1})D] \\ &+ \psi_j[(x_0 - x_{j+1})D + (y_0 - y_{j+1})C]\} \end{aligned} \quad (14b)$$

where

$$C = [A(x_{j+1} - x_j) + B(y_{j+1} - y_j)]/F \quad (14c)$$

$$D = [B(x_{j+1} - x_j) - A(y_{j+1} - y_j)]/F \quad (14d)$$

$$F = (x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2 \quad (14e)$$

$$A = \ln\left|\frac{(z_{j+1} - z_0)}{(z_j - z_0)}\right| \quad (14f)$$

$$B = \theta(z_{j+1}, z_j; z_0) \quad (14g)$$

Equations (14a) and (14b) can be rewritten as

$$\begin{aligned}2\pi\hat{\phi}(z_0) = \sum_{j=1}^{N} \{&x_0[C(\psi_{j+1} - \psi_j) + D(\phi_{j+1} - \phi_j)] \\ &+ y_0[D(\psi_j - \psi_{j+1}) + C(\phi_{j+1} - \phi_j)] \\ &+ [-\psi_j(-x_{j+1}C + y_{j+1}D) \\ &- \phi_j(-x_{j+1}D - y_{j+1}C) \\ &+ \psi_{j+1}(-x_jC + y_jD) \\ &+ \phi_{j+1}(-x_jD - y_jC)]\} \end{aligned} \quad (15a)$$

and

$$\begin{aligned}-2\pi\hat{\psi}(z_0) = \sum_{j=1}^{N} \{&x_0[D(\psi_j - \psi_{j+1}) + C(\phi_{j+1} - \phi_j)] \\ &+ y_0[C(\psi_j - \psi_{j+1}) + D(\phi_j - \phi_{j+1})] \\ &+ [\psi_j(-x_{j+1}D - y_{j+1}C) \\ &- \phi_j(-x_{j+1}C + y_{j+1}D) \\ &- \psi_{j+1}(-x_jD - y_jC) \\ &+ \phi_{j+1}(-x_jC + y_jD)]\} \end{aligned} \quad (15b)$$

The terms in braces can be expressed, compactly as

$$C_{1,j} = C(\psi_{j+1} - \psi_j) + D(\phi_{j+1} - \phi_j) \quad (16)$$

$$C_{2,j} = D(\psi_j - \psi_{j+1}) + C(\phi_{j+1} - \phi_j) \quad (17)$$

$$\begin{aligned}C_{3,j} = &-\psi_j(-x_{j+1}C + y_{j+1}D) \\ &- \phi_j(-x_{j+1}D - y_{j+1}C) \\ &+ \psi_{j+1}(-x_jC + y_jD) + \phi_{j+1}(-x_jD - y_jC) \end{aligned} \quad (18)$$

$$\begin{aligned}C_{4,j} = &\psi_j(-x_{j+1}D - y_{j+1}C) - \phi_j(-x_{j+1}C + y_{j+1}D) \\ &- \psi_{j+1}(-x_jD - y_jC) + \phi_{j+1}(-x_jC + y_jD) \end{aligned} \quad (19)$$

Substituting equations (16)–(19) into equations (15a) and (15b) yields

$$x_0\sum_{j=1}^{N} C_{1,j} + y_0\sum_{j=1}^{N} C_{2,j} = 2\pi\hat{\phi}(z_0) - \sum_{j=1}^{N} C_{3,j} \quad (20)$$

and

$$x_0\sum_{j=1}^{N} C_{2,j} - y_0\sum_{j=1}^{N} C_{1,j} = -2\pi\hat{\psi}(z_0) - \sum_{j=1}^{N} C_{4,j} \quad (21)$$

With the domain geometry specified and the values of ϕ and ψ known at interior point z_0 (the hats can then be dropped) and at the boundary nodes, equations (20) and (21) become two simultaneous

equations in two unknowns: x_0 and y_0. Unfortunately, these equations are nonlinear due to the presence of terms A and B in the summations. Recall that A and B contain $z_0 = x_0 + iy_0$ (see equations [14f] and [14g]).

It has been found that a single-point iterative solution strategy is effective for solving these nonlinear equations simultaneously for x_0 and y_0. To facilitate the implementation of this process, equations (20) and (21) are combined to yield

$$y_0 = \frac{\left\{\dfrac{2\pi\phi(z_0) - \sum_{j=1}^{N} C_{3,j}}{\sum_{j=1}^{N} C_{1,j}} + \dfrac{2\pi\psi(z_0) - \sum_{j=1}^{N} C_{4,j}}{\sum_{j=1}^{N} C_{2,j}}\right\}}{\left\{\dfrac{\sum_{j=1}^{N} C_{2,j}}{\sum_{j=1}^{N} C_{1,j}} + \dfrac{\sum_{j=1}^{N} C_{1,j}}{\sum_{j=1}^{N} C_{2,j}}\right\}} \tag{22}$$

and

$$x_0 = \frac{2\pi\phi(z_0) - \sum_{j=1}^{N} C_{3,j}}{\sum_{j=1}^{N} C_{1,j}} - \frac{\sum_{j=1}^{N} C_{2,j}}{\sum_{j=1}^{N} C_{1,j}} y_0 \tag{23}$$

The process of solving for x_0 and y_0 with $\phi(z_0)$ and $\psi(z_0)$ known proceeds as follows:

(1) Guess initial values for x_0 and y_0;
(2) Calculate $C_{1,j}$, $C_{2,j}$, $C_{3,j}$, and $C_{4,j}$ using the values of x_0 and y_0;
(3) Calculate a new value of y_0 using equation (22);
(4) Calculate a new value of x_0 using equation (23); the values of $C_{1,j}$, $C_{2,j}$, $C_{3,j}$, and $C_{4,j}$ are taken from Step 2, while the new value of y_0 from Step 3 is used; and
(5) Repeat Steps 2 through 5 until the percent difference between the new and previous values of x_0 and y_0 is less than a predetermined constant, ε.

These five steps comprise the desired iterative process for calculating the location of an interior point when the values of ϕ and ψ are known at the boundary nodes and at the point itself. This process is now combined with the linear element CVBEM to create a method for generating grid systems in 2D simply connected spatial domains. The method will henceforth be referred to as the complex variable boundary element grid generation method or CVBEGGM for short. The fundamental concept behind the CVBEGGM is the use of potential lines and streamlines as grid lines, and it is intended for domains whose boundaries can be divided into four separate, continuous, smooth, or nonsmooth curves.

The CVBEGGM consists of the following eight steps:

Step 1: *Specify the domain geometry.* Prescribe the locations of the boundary-element nodal points along the domain boundary. The number of nodal points will have a direct effect on the accuracy of the solution (i.e., the quality of the grid). As more nodal points are used, the grid quality improves, but the amount of time required to generate the grid increases as well. There is, therefore, a compromise to be made. Certainly, the number of nodal points should be large enough to adequately describe the boundary contour itself but need not be equal to the number of grid points ultimately desired along the boundaries.

Step 2: *Define the mapping.* The CVBEGGM is intended to map a 2D simply connected, physical spatial domain onto a 2D, rectangular computational domain. The boundary of the physical domain will be mapped one-to-one to the four sides of the computational domain, forming a so-called body-fitted coordinate system.[4] Since the grid lines of each family (ξ and η) connect opposing sides of the rectangular computational domain (see *Figure 1*), the boundary of the physical domain needs to be broken down into four separate boundary curves (see *Figure 3*). Reference 4 gives a good discussion of various breakup considerations for simply connected domains.

Step 3: *Apply the boundary conditions.* The boundary conditions that should be imposed to ensure that the grid lines intersect the boundaries orthogonally are shown in *Figure 3*. These boundary conditions force boundary curves 1 and 3 to be streamlines and boundary curves 2 and 4 to be potential lines. Streamlines in the domain will thus connect and intersect orthogonally boundary curves 2 and 4, while potential lines in the domain will connect and intersect orthogonally boundary curves 1 and 3. The streamlines and potential lines will become grid lines and will be mapped to lines of constant η and ξ, respectively, in the computational domain. These grid lines will intersect both each other and the domain boundaries orthogonally, excellent characteristics for a grid system to possess.[4] The corner nodes should be treated using the following rules:

(a) the corner node between boundary curves 4 and 1 is treated as a completely specified node ($\phi = 0, \psi = 0$);
(b) the corner node between boundary curves 1 and 2 is treated as a completely specified node ($\phi = 1, \psi = 0$);

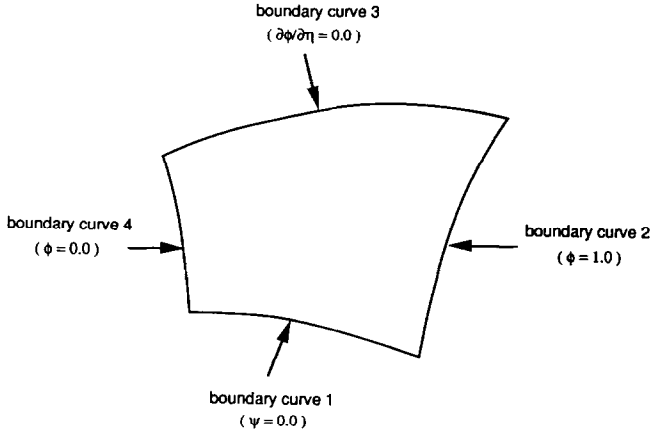

Figure 3. The boundary break-up and boundary conditions used in the CVBEGGM.

(c) the corner node between boundary curves 2 and 3 is treated by specifying $\phi = 1$ directly and applying equation (11) to solve for $\hat{\psi}$ and
(d) the corner node between boundary curves 3 and 4 is treated by specifying $\phi = 0$ directly and applying equation (9) to solve for $\hat{\psi}$.

Step 4: *Solve for the unknown values of ϕ and ψ at the boundary nodes.* This is accomplished using the standard linear-element CVBEM as described previously.

Step 5: *Calculate the boundary grid point locations.* There are a number of ways of specifying the boundary grid point locations, but in the most basic approach, one first prescribes equally spaced grid points along boundary curves 1 and 4. The locations of these grid points are calculated by approximating the arc length along the boundary as the cumulative sum of the lengths of the linear segments connecting the nodal points. Linear spline interpolation is then used to determine the equally spaced positions. The number of grid points along curve 1 (and along streamlines) will be designated as IL, while the number of grid points along curve 4 (and along potential lines) will be designated as JL. Grid point locations will be given by $[x(i, j), y(i, j)]$ where i runs from 1 to IL and j runs from 1 to JL; see the example in the physical domain shown in *Figure 4*. Notice that, in general, the boundary grid points need not coincide with the nodal points except at the corners.

Having chosen the grid point locations along the boundary curves 1 and 4, one then calculates the values of either ϕ or ψ at each of these points—ϕ for the points on curve 1 and ψ for the points on curve 4—using linear interpolation between the nodal point values *calculated* by the CVBEM.

Finally, the values of ϕ and ψ at the grid points along boundary curves 1 and 4 are used to calculate the grid point locations along boundary curves 2 and 3. Because of the boundary conditions imposed, each grid point on boundary curve 1 has a corresponding grid point on boundary curve 3 that has the same value of ϕ. Likewise, each grid point on boundary curve 4 has a corresponding grid point on boundary curve 2 that has the same value of ψ. For each grid point on boundary curves 1 and 4, the corresponding grid point on boundary curves 2 and 3 is located by using linear interpolation between the values of ϕ or ψ (ψ for curve 2 and ϕ for curve 3) at the nodal points.

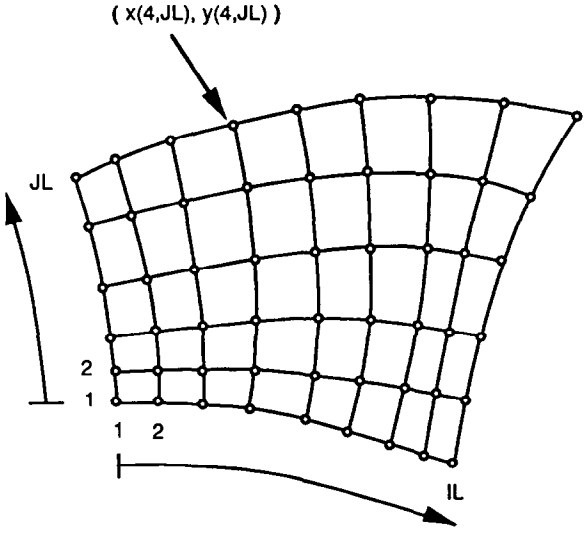

Figure 4. Grid point locations along the boundaries of the domain.

Step 6: *Calculate the initial interior grid point locations.* There are several ways that an initial distribution of interior grid points can be specified; here, it is recommended that transfinite bilinear interpolation be used.[15] This method is quite computationally efficient and has been found to provide acceptable initial guesses for the interior grid point locations. The equations for calculating the initial interior grid point locations using this method are

$$\begin{aligned} x(i,j) = {} & (1-\eta)x(i,1) + \eta x(i,\mathrm{JL}) + (1-\xi)x(1,j) \\ & + \xi x(\mathrm{IL},j) - [(1-\xi)(1-\eta)x(1,1) \\ & + \xi(1-\eta)x(\mathrm{IL},1) + (1-\xi)\eta x(1,\mathrm{JL}) \\ & + \xi\eta x(\mathrm{IL},\mathrm{JL})] \end{aligned} \tag{24}$$

and

$$\begin{aligned} y(i,j) = {} & (1-\eta)y(i,1) + \eta y(i,\mathrm{JL}) + (1-\xi)y(1,j) \\ & + \xi y(\mathrm{IL},j) - [(1-\xi)(1-\eta)y(1,1) \\ & + \xi(1-\eta)y(\mathrm{IL},1) + (1-\xi)\eta y(1,\mathrm{JL}) \\ & + \xi\eta y(\mathrm{IL},\mathrm{JL})] \end{aligned} \tag{25}$$

where

$$\xi = \frac{i-1}{\mathrm{IL}-1} \qquad \eta = \frac{j-1}{\mathrm{JL}-1}$$

and

$$i = 2, 3, \ldots, \mathrm{IL}-1 \qquad j = 2, 3, \ldots, \mathrm{JL}-1$$

Step 7: *Calculate the final interior grid point locations.* This is accomplished via the five-step iterative procedure described previously. Each interior grid point location is calculated using this iterative procedure. It has been found that, for a convergence criteria of $\varepsilon = 0.001$, convergence is usually obtained within two to three iterations.

Step 8: *Calculate the grid point locations in the computational domain.* Having calculated the locations of the grid points in the physical domain, it is now necessary to calculate their mapped locations in the computational domain. This operation could have been performed first, but it is sufficient to do so now. The locations of the grid points in the computational domain are given by the ordered pairs (ξ_i, η_j) where

$$\xi_i = (i-1)\Delta\xi \qquad i = 1, 2, \ldots, \mathrm{IL}$$

$$\eta_j = (j-1)\Delta\eta \qquad j = 1, 2, \ldots, \mathrm{JL}$$

and

$$\Delta\xi = \frac{1}{\mathrm{IL}-1} \qquad \Delta\eta = \frac{1}{\mathrm{JL}-1}$$

3.1 *Additional comments on the CVBEGGM*

This subsection is included in order to mention some of the more subtle points associated with the CVBEGGM. It was stated in Step 1 of the method that the number of nodal points used can affect the quality of the resulting grid. If the boundaries are composed mostly of flat segments, then very few nodal points may be required to describe the domain geometry. This does not mean that few nodal points are required to produce an acceptable grid, however, since the values of ϕ and ψ may deviate substantially from linearity over a geometrically linear boundary. The linear approximation associated with the method will introduce errors at the boundaries of the domain. Increasing the number of nodal points increases the accuracy of the solution, thus improving smoothness and near-orthogonality. However, it also increases the number of simultaneous equations that must be solved, as mentioned previously. The recommended direct method of solution (Gaussian elimination with scaled partial pivoting) will be subject to roundoff errors when the number of equations (nodes) becomes "large". ("Large" will be machine- and compiler-dependent.) Thus, care should be exercised when increasing the number of nodal points to verify that the solution is actually improved. An example showing the effects of increasing the number of nodal points is given in the next section.

As described in Step 5, the CVBEGGM allows the specification of grid point locations along two of the four boundary curves. The grid point locations along the other two curves are fixed by this specification. Thus, the potential exists for generating two very different grids simply by reversing the boundaries along which grid point locations are specified. If the domain of interest has important boundary features on all four sides, then the specified boundary grid point locations need to be selected carefully to ensure that the grid points on the remaining two boundaries adequately capture the domain geometry.

Another point worthy of note involves the distribution of grid points in the physical domain. Many traditional elliptic PDE grid generation methods employ Poisson's equation, utilizing the source term to control the distribution of grid lines and grid points.[4] Since the CVBEM is capable of solving Poisson's equation in certain cases, the use of this approach in the CVBEGGM is being investigated; however, some measure of grid line control can be achieved by simply varying the boundary grid point distribution. Instead of prescribing equally spaced grid points along boundaries 1 and 4 (as described in Step 5), one can vary the spacing using 1D stretching functions. A list of such functions is given in Ref. 1. This approach has limitations, however, since due to the smoothing characteristics of the Laplacian, grid lines tend to be more equally spaced away from the boundaries regardless of the stretching function used.[4]

4. Examples and results

The capabilities of the CVBEGGM are now demonstrated by application of the method to five 2D simply-connected spatial domains.

Example Domain 1 (see *Figure 5*) is a region bounded by three curved, concave segments and one linear segment. The grid generated by the CVBEGGM for this domain is 21 × 21, and 38 nodal points were used to define the boundary. One can see that the grid lines are smooth and that they are nearly orthogonal to each other and to the boundaries. The variation in spacing of the grid lines is a natural consequence of the method due to the solution of Laplace's equation.[4] Grid lines are seen to be less closely spaced over the concave boundaries than over the straight boundary.

It was mentioned that 1D stretching functions could be used to control the grid-point distribution in the physical domain. An example showing the results of such a procedure is presented in *Figure 6*. The geometry is that of Example Domain 1, and the grid lines have been

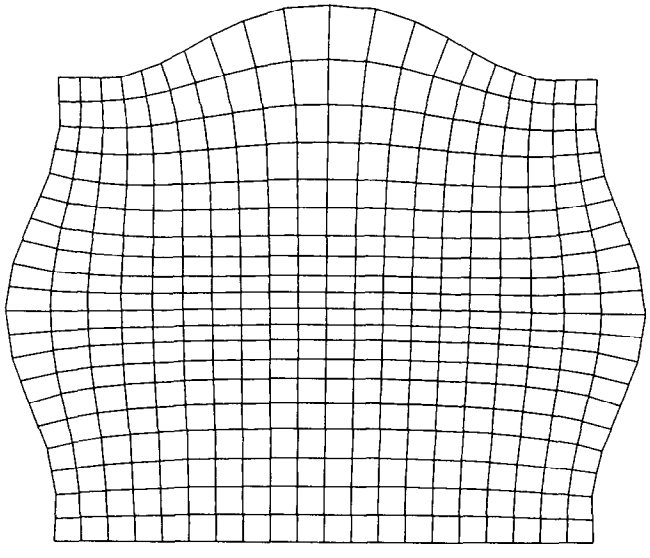

Figure 5. Grid generated by the CVBEGGM for Example Domain 1.

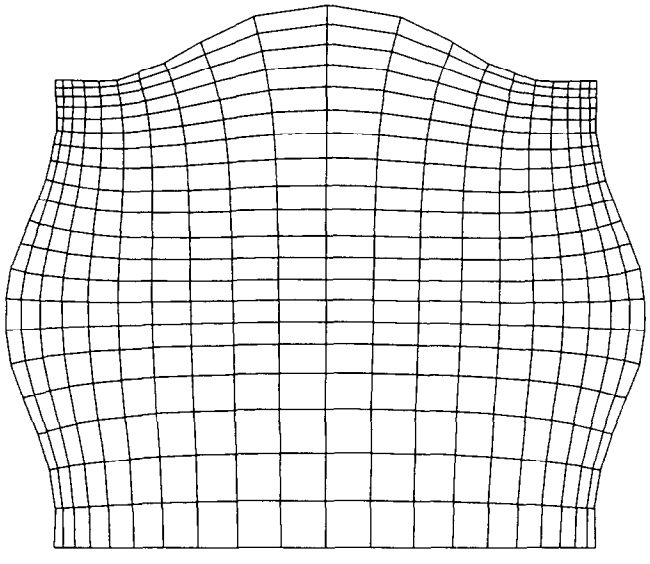

Figure 6. Grid from *Figure 5* with stretching functions used.

clustered near the left, right, and upper boundaries by using stretching functions given in Ref. 1. This redistribution of grid points was accomplished without resolving the associated system of simultaneous equations. (Only Steps 5, 6, and 7 of the CVBEGGM were performed.)

Example Domain 2 (see *Figures 7* and *8*) is a trapezoidal region. Two 15 × 15 grids were generated for this domain using the CVBEGGM: one using 16 nodal points (*Figure 7*) and the other using 56 nodal points (*Figure 8*). Notice that in *Figure 7*, the grid lines do not intersect the domain boundaries orthogonally. This is attributed to the relatively small number of nodal points (16), leading to a poor approximation of ϕ and ψ at the boundaries. In contrast, the grid lines in *Figure 8* do intersect the domain boundaries with near-orthogonality. These two figures demonstrate the effect that the number of nodal points used can have on the grid quality.

The grid in *Figure 9* (Example Domain 3) represents a cylinder in cross flow, and 105 boundary nodes were used to generate this 25 × 31 grid. This example demonstrates the ability of the method to generate a smooth grid even when the domain boundary possesses a large slope discontinuity. The shock-like compaction that propagates from the cylinder corner is another example of the spacing characteristics of the method. This compaction probably represents an inefficient concentration of grid points, and by clustering grid points near the left and right boundaries, the compaction can be lessened.

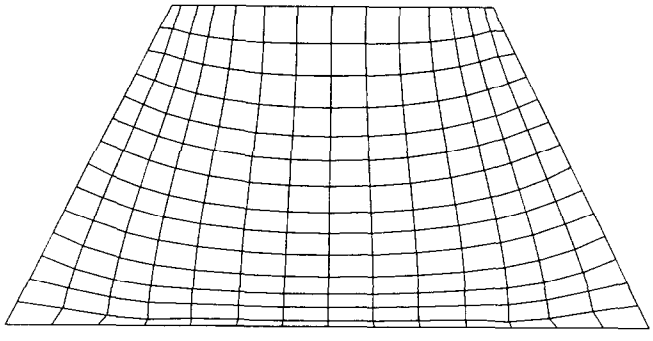

Figure 7. Grid generated by the CVBEGGM for Example Domain 2 using 16 nodal points.

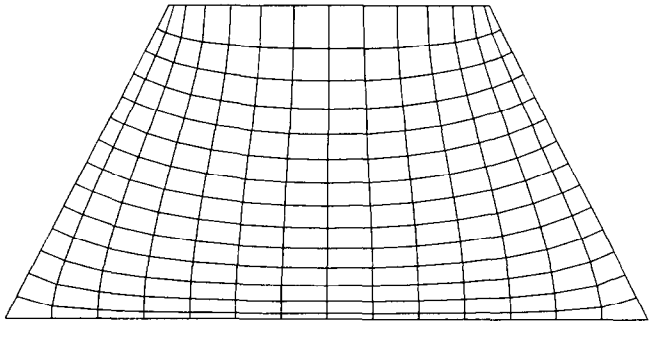

Figure 8. Grid generated by the CVBEGGM for Example Domain 2 using 56 nodal points.

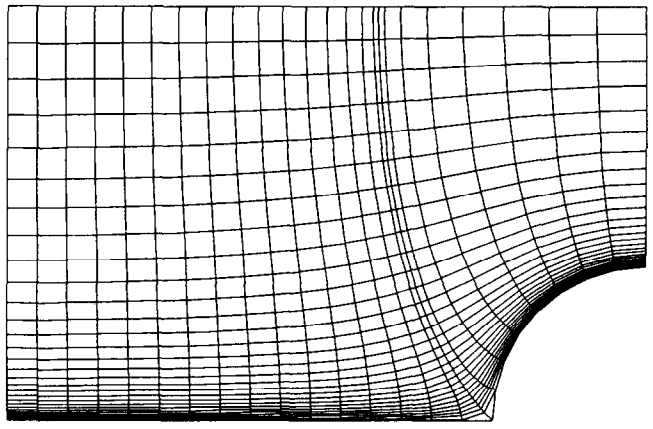

Figure 9. Grid generated by CVBEGGM for Example Domain 3.

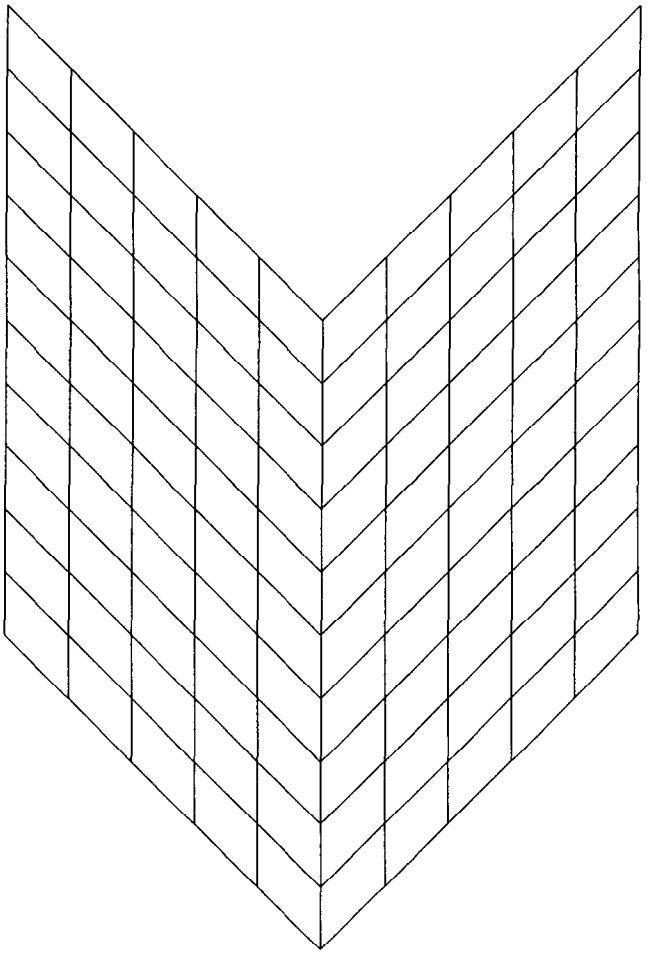

Figure 10. Grid generated for Example Domain 4 using bi-linear TFI.

Example Domain 4, the chevron, is a geometry that presents unique challenges to grid generation methods.[16] Like Example Domain 3, the boundaries of the domain possess slope discontinuities. Additionally, the upper and lower surfaces are strongly convex and concave, respectively. For comparison purposes, grids generated using bilinear transfinite interpolation (TFI, an algebraic method[15]), Winslow's elliptic method,[16] and the CVBEGGM are presented in *Figures 10*, *11*, and *12*,

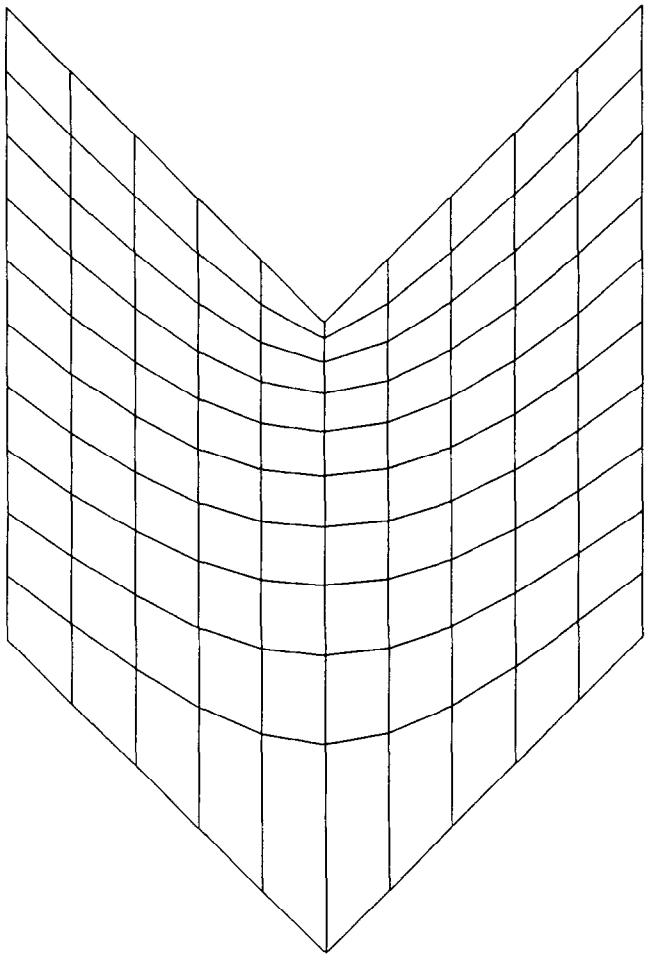

Figure 11. Grid generated for Example Domain 4 using Winslow's elliptic method.

respectively. One can see that in the TFI grid (*Figure 10*), the boundary slope discontinuity has propagated into the grid interior, and the grid lines do not intersect the boundaries orthogonally. The grid generated using Winslow's method (*Figure 11*) does not have the slope discontinuity propagation, but the grid line orthogonality is again not present at the boundaries. The CVBEGGM grid (*Figure 12*) exhibits both desirable characteristics, namely, (1) no propagation of the boundary slope discontinuity into the grid interior and (2) grid line near-orthogonality at the domain boundaries. Such orthogonality can be achieved using Winslow's method with the appropriate selection of source terms, but the process is computationally intensive and difficult to achieve at all boundaries simultaneously.[16]

In order to demonstrate that grids constructed by the CVBEGGM are suitable for obtaining numerical solutions to fluid flow problems, the grid shown in *Figure 13* was used by an FDM[17] to generate a viscous-flow solution. This grid represents a converging-diverging nozzle (Example Domain 5). The clustering of grid lines near the nozzle walls was accomplished using stretching functions, and the 31 × 41 grid was generated using 84 nodal points along the boundary. The resulting pressure contours and velocity vectors are shown in *Figure 14*. For this example, a Reynolds number of 500 was chosen in order to ensure laminar flow and avoid having to introduce a turbulence model. The development of the boundary layer along the nozzle walls can be clearly observed as can the acceleration and subsequent deceleration of the flow. The maximum velocity at the center of the throat is 2.44 times the inlet velocity, and the centerline exit velocity is 1.74 times the inlet velocity. It is noted that the flow field exhibits all of the characteristics expected to be associated with the geometry investigated.

All of the grids presented herein were generated by using a 66 MHz 80486DX2-based personal computer. As

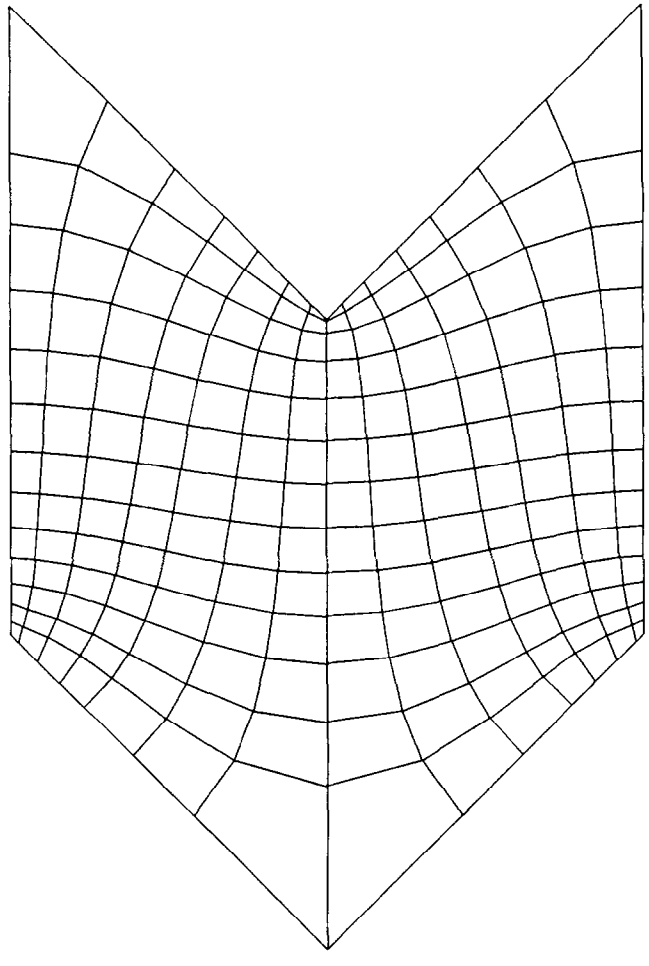

Figure 12. Grid generated for Example Domain 4 using the CVBEGGM.

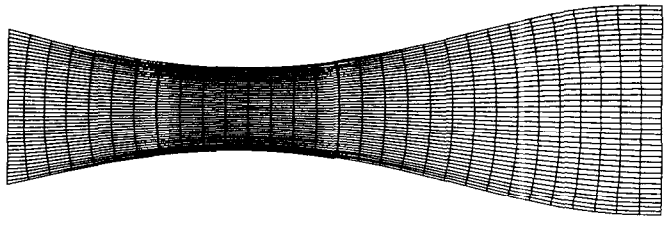

Figure 13. Grid generated for Example Domain 5 using the CVBEGGM.

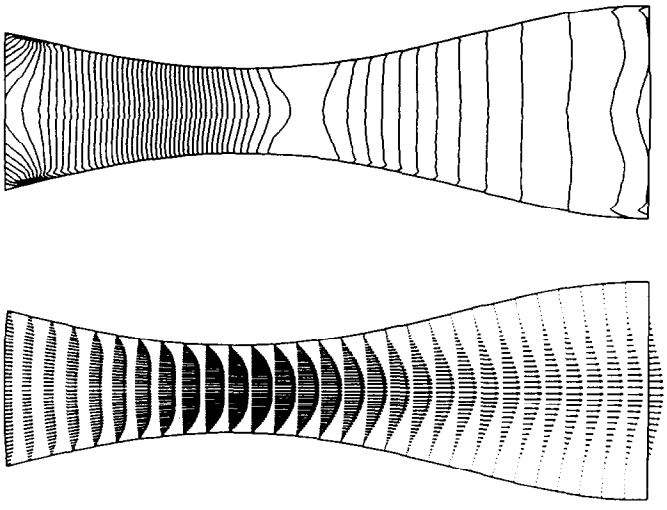

Figure 14. The (a) pressure contours and (b) velocity vectors calculated by an FDM using the grid shown in *Figure 13*.

an indication of the efficiency of the method, the grid shown in *Figure 5* took 8 sec to generate.

5. Summary and conclusions

A variation of the CVBEM was applied in the area of numerical grid generation. This method, dubbed the complex variable boundary element grid generation method (CVBEGGM), can be used to generate grid systems for 2D, simply connected spatial domains. The method consists of two primary phases. First, the boundary of the domain is imposed with specific boundary conditions, and the linear element CVBEM is used to calculate the unknown values of ϕ and ψ along the boundary. Second, an iterative search is used to determine the positions of the boundary and interior points that are located at the intersections of designated streamlines and potential lines. These points are then used as grid points. The method allows one to control the grid point distribution directly over two of the four boundary curves and to enforce grid line near-orthogonality at both the interior and the boundary of the domain. The method is similar to the elliptic PDE grid generation methods in that Laplace's equation is solved, but since a boundary element method is used, refinement of the grid can be achieved without resolving the associated system of simultaneous equations.

The CVBEGGM was tested via application to five 2D four-sided simply connected spatial domains. It was found that the grid lines generated were smooth and that they intersected both each other and the domain boundaries with near-orthogonality. One-dimensional stretching functions were observed to be effective in controlling the distribution of grid points, although some natural variation in the spacing is inherent in the method due to the characteristics of the Laplace equation. The 8-sec computation time for a 21 × 21 grid on a 66 MHz 80486DX2-based personal computer indicates the computational efficiency of the method.

A viscous flow solution was generated by a finite-difference based code using one of the grids. The solution obtained satisfactorily captured the flow characteristics associated with the specific geometry investigated.

References

1 Anderson, D. A., Tannehill, J. C. and Pletcher, R. H. *Computational Fluid Mechanics and Heat Transfer*. Hemisphere Publishing Corp., Washington, D.C., 1984

2 Hoffmann, K. A. and Chiang, S. T. *Computational Fluid Dynamics for Engineers*. Volumes I and II, Engineering Education System, Wichita, Kansas, 1993

3 Winslow, A. M. Equipotential zoning of two-dimensional meshes. *J. Comput. Phys.* 1966, **1**, 149–165

4 Thompson, J. F., Warsi, Z. U. A. and Mastin, C. W. *NumericalGrid Generation Foundations and Applications*. North-Holland, New York, 1985

5 Thompson, J. F. and Warsi, Z. U. A. Boundary-fitted coordinate systems for numerical solution of partial differential equations—A review. *J. Comput. Phys.* 1982, **47**, 1–108

6 Thompson, J. F. Grid generation techniques in computational fluid dynamics. *AIAA J.* 1984, **22**, 1505–1510

7 Thompson, J. F. Composite grid generation code for general 3-D regions—The EAGLE code. *AIAA J.* 1988, **26**(3), 271–272

8 Banerjee, P. K. and Butterfield, R. *Boundary Element Methods in Engineering Science*. McGraw-Hill Book Co. (UK) Limited, London, 1981

9 Brebbia, C. A. and Dominguez, J. *Boundary Elements An Introductory Course*. Computational Mechanics Publications, Boston, Massachusetts, and McGraw-Hill Book Co., New York, New York, 1992

10 Tsay, T.-K. Grid generation by using boundary integral element method. *Numerical Grid Generation in Computational Fluid Mechanics '88: Proceedings of the Second International Conference*. Pineridge Press, Ltd., Swansea, Wales, 1988, pp. 23–30

11 Hromadka II, T. V. and Lai, C. *The Complex Variable Boundary Element Method in Engineering Analysis*. Springer-Verlag New York, Inc., New York, 1987

12 Bailey, R. T. Extensions and refinements to the complex variable boundary element method including its application to numerical grid generation. Ph.D. Thesis, University of Florida, Gainesville, FL, 1991

13 Kassab, A. J. and Hsieh, C. K. Application of the complex variable boundary element method to solving potential problems in doubly connected domains. *Int. J. Numer. Meth. Eng.* 1990, **29**, 161–179

14 Kassab, A. J. and Chesla, S. CVBEM solution of non-linear heat conduction problems. *Proceedings of the Seventh International Conference on Boundary Element Technology*, ed. C. A. Brebbia and M. S. Ingber, Computational Mechanics Publications, Boston, 1992, pp. 457–472

15 Coons, S. A. Surfaces for computer-aided design of space forms. MIT-MAC-TR-41, Cambridge, MA, 1967

16 Knupp, P. and Steinberg, S. *Fundamentals of Grid Generation*. CRC Press, Ann Arbor, MI, 1993

17 Li, H. The efficient numerical solution of two-dimensional compressible inviscid and viscous flow of advanced internal and external flow systems. M.S. Thesis, University of Florida, Gainesville, FL, 1991